REMARKS

Claims 1-11 were pending in the parent application. In the most recent Office Action, Claims 1-9 were rejected and Claims 10-11 were allowed. Claims 1, 5 and 8 have been amended. Claim 10 has been cancelled and Claim 11 from the parent case is not present in the continuation. New Claims 11-23 have been added. Thus, Claims 1-9 and 11-23 are now pending in the application.

Section 103 Rejections

The Examiner rejected Claims 1-4 from the parent case under 35 U.S.C. § 103(a) as being unpatentable over Heizer (U.S. Patent No. 5,249,290) (Heizer) in view of Thacker (U.S. Patent No. 5,267,235) (Thacker). Applicants respectfully traverse this rejection for at least the following reasons.

Heizer involves using server processes to access shared server resources in response to service requests from client computers connected to the network. Heizer, Abstract. Thacker involves presenting requests synchronously to all servers to which access is desired. Thacker, Abstract. In a grant phase, Thacker notes that each server simultaneously examines all access requests and selects precisely one such request, to whose requester a grant is issued. Thacker, col. 2, lines 60-64. Then, in an acceptance phase, each requester simultaneously examines all grants sent to it by the servers and selects one such grant. Thacker, col. 3, lines 1-4.

The combination of Heizer and Thacker is improper because the combination of Thacker into Heizer would change the principle of operation of Heizer and render Heizer inoperative. MPEP 2143.01. Heizer notes that a control process at the server decides which of the server processes should handle the client and that server processes receive a client request, process the request and send a response. Heizer, col. 3, lines 10-24. Thacker relies on the requester to determine which server will handle the request. Thacker, col. 3, lines 1-4. Heizer's server-side control of service would fail to operate with Thacker's requester-side control of service because Heizer places control at a different location than Thacker. In addition, there would be no reasonable expectation of success for the combination of Heizer and Thacker because Heizer and Thacker place control at separate locations. Also, efficiency would decrease because the combination Heizer and Thacker would require that both the client and server determine which server processes are to service which client requests.

Therefore, for at least these reasons, the combination of Heizer and Thacker is improper. Thus, Applicants respectfully request withdrawal of this combination.

Neither Heizer nor Thacker, either alone or in combination, teach or suggest every element of amended Claim 1. Amended Claim 1 recites, in part, "a plurality of handler processes associated with the server system and available to service pending client requests, the handler processes being operable to access a notification system in parallel", "accept pending requests in parallel", and such that "one of which will service each pending request when the number of handler processes exceeds the number of pending requests and to accept a number of pending requests substantially equal to the number of handler processes when the number of pending requests exceeds or equals the number of handler processes." The Examiner admits that Heizer "does not teach the resource arbitration includes notifying the plurality of handler processes in parallel, attempting to accept requests in parallel, one of the handler processes servicing one request and remaining processing other requests" and relies solely on Thacker to teach these elements. Office Action, p. 2. Thacker places the final decision of which grant will service which request at the requester, in contrast to the recitation in amended Claim 1 of "a plurality of handler processes associated with the server system".

Also, Thacker uses multiple iterations in matching up requesters and servers by using a grant phase and an acceptance phase. Thacker, col. 4, lines 28-44. In contrast, Claim 1 recites "one of which will service each pending request when the number of handler processes exceeds the number of pending requests and to accept a number of pending requests substantially equal to the number of handler processes when the number of pending requests exceeds or equals the number of handler processes." [emphasis added] Thacker teaches away from Claim 1's handling of each request because Thacker uses a multiple iteration process to handle requests. Indeed, Thacker averages only about a 75% acceptance rate. Thacker, col. 4, lines 34-36.

Therefore, for at least these reasons, Applicants respectfully submit that Claim 1 is patentable over the cited references of Heizer and Thacker. Thus, Applicants respectfully request allowance of Claim 1.

The Examiner rejected Claims 5-8 in the parent case under 35 U.S.C. § 103(a) as being unpatentable over Heizer (U.S. Patent No. 5,249,290) (Heizer) in view of Thacker

(U.S. Patent No. 5,267,235) (Thacker) as applied to Claim 1 and further in view of Furtney et al (U.S. Patent No. 5,257,372) (Furtney). Applicants respectfully traverse this rejection for at least the following reasons.

Independent Claim 5 is allowable at least for the reasons discussed above. Furtney is relied upon by the Examiner only to teach "awakening handler processes" and the "parallel client server system." Office Action, p. 3. The Examiner does not rely upon Furtney, nor does Furtney teach, the elements discussed above in association with Claim 1. In particular, Claim 5 recites "creating a plurality of handler processes with a spawner process at a server", "notifying, in parallel, a plurality of the handler processes that at least one request has arrived", "accepting each pending request from the buffer, in parallel, with the plurality of handler processes when the number of handler processes exceeds the number of pending requests", and "accepting a number of pending requests substantially equal to the number of handler processes when the number of pending requests exceeds or equals the number of handler processes." Accordingly, Applicants respectfully request that Claim 5 be allowed.

Dependent Claims 2-4, which depend from independent Claim 1, and dependent Claims 6-9, which depend from independent Claim 5, are also allowable for the same reasons as their respective base claims as defining further distinctions over the cited references. Accordingly, Applicants respectfully request allowance of these dependent claims.

New Claims

Applicants have added new Claims 11-23. Applicants respectfully submit that no new matter has been added by new Claims 11-23. New independent Claims 11-12, 17 and 23 are allowable at least for the reasons discussed above. Accordingly, Applicants respectfully request that Claims 11-12, 17 and 22 be allowed.

In particular, Claim 11 recites "means for creating a plurality of handler processes with a spawner process at a server", "means for notifying, in parallel, a plurality of the handler processes that at least one request has arrived", "means for accepting each pending request from the buffer, in parallel, with the plurality of handler processes when the number of handler processes exceeds the number of pending requests" and "means for accepting a number of pending requests substantially equal to the number of handler processes when the number of pending requests exceeds or equals the number of handler processes." Claim 12 recites "providing, at a server, at least one available handler process, the available handler process comprising a handler process which is not presently processing a previously accepted

pending request", "notifying, in parallel, the handler processes that at least one pending request is in the buffer", "accepting substantially all pending requests from the buffer, substantially in parallel, with the plurality of handler processes, when a number of pending requests is less than or equal to a number of available handler processes" and "accepting a number of pending requests from the buffer substantially equal to the number of available handler processes when the number of pending requests is greater than the number of available handler processes". Claim 17 recites "software encoded on a computer readable medium, the software operable to:", "provide, at a server, at least one available handler process, the available handler process comprising a handler process which is not presently processing a previously accepted pending request", "notify, in parallel, the handler processes that at least one pending request is in the buffer", "accept substantially all pending requests from the buffer, substantially in parallel, with the plurality of handler processes, when a number of pending requests is less than or equal to a number of available handler processes", and "accept a number of pending requests from the buffer substantially equal to the number of available handler processes when the number of pending requests is greater than the number of available handler processes". Claim 22 recites "means for providing, at a server, at least one available handler process, the available handler process comprising a handler process which is not presently processing a previously accepted pending request", "means for notifying, in parallel, the handler processes that at least one pending request is in the buffer", "means for accepting substantially all pending requests from the buffer, substantially in parallel, with the plurality of handler processes, when a number of pending requests is less than or equal to a number of available handler processes" and "means for accepting a number of pending requests from the buffer substantially equal to the number of available handler processes when the number of pending requests is greater than the number of available handler processes".

Dependent Claims 13-16 that depend from new independent Claim 12, dependent Claims 18-21 that depend from new independent Claim 17, and dependent Claim 23 that depends from independent Claim 5 are also allowable for at least the same reasons as their respective base claims and as defining further distinctions over the cited references. Accordingly, Applicants respectfully submit that these dependent claims be allowed.

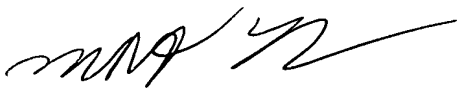
CONCLUSION

Applicants have made an earnest attempt to place this case in condition for allowance. For the foregoing reasons and for other reasons clearly apparent, Applicants respectfully request reconsideration and full allowance of all pending claims.

If the Examiner believes a telephone conference would advance prosecution of this case, please call the undersigned attorney for Applicants at 214-953-6984.

Applicant has enclosed a check in the amount of $986.00 to cover the filing fee for the new claims. The Commissioner is hereby authorized to charge any additional fees or credit any overpayments to Deposit Account No. 02-0384 of BAKER BOTTS L.L.P.

Respectfully submitted,
BAKER BOTTS L.L.P.
Attorneys for Applicants



Matthew B. Talpis
Reg. No. 45,152

Correspondence Address:
Matthew B. Talpis, Esq.
Baker Botts L.L.P.
2001 Ross Avenue, Suite 600
Dallas, Texas 75201-2980
Phone: (214) 953-6984
Fax: (214) 661-4984
matt.talpis@bakerbotts.com

Date: August 22, 2001

MARKED UP VERSION OF CLAIM AMENDMENTS

IN THE SPECIFICATION

On Page 3, lines 7-19, please replace the paragraph as follows:

--The pioneers in the development of client-server systems, **[use to the high degree of control available in traditional single-program single-computer systems]**, developed client-server communication systems that were biased toward the processing of competing client requests for a single published service in a serial manner. This serial processing method of design, developed from the single-program single-computer paradigm, led to the creation of bottlenecks in the client-server communication process. The bottleneck in the client-server communication process typically appeared between the arrival of the client request at a server and the dispatching of the received request for service.--

IN THE CLAIMS

1. A software system comprising:

a server system comprising an operating system, the operating system operable to support a well-known address, the well-known address operable to receive data, the operating system further operable to provide interprocess communication;

the operating system further operable to support a buffer associated with the well-known address, the buffer **[comprising a finite amount of memory, and]** operable to store data received by the well-known address;

a plurality of handler processes **associated with the server system and available to service pending client requests, the handler processes being** operable to access a notification system in parallel, **[to attempt parallel acceptance of]** **accept** pending requests in parallel **[to process error conditions]**, and to provide service to client requests, such that **[a single]** **at least one** request received by the well-known address will result in the notification to a plurality of **the** handler processes, one of which will service **[the]** **each pending** request **[, the remaining handler processes are operable to service other, later requests or to perform error processing if no other requests are present]** **when the number of handler processes exceeds the number of pending requests and to accept a number of pending requests substantially equal to the number of handler processes when the number of pending requests exceeds or equals the number of handler processes**;

the operating system further comprising **[a notification system, the notification system may comprises any suitable hardware and/or software]** **the notification system, the notification** system operable to be accessed by **[a]** **the** **[plurality of]** handler processes **[, at any time]**, the notification system further operable to reflect the existence **[or]** **of data in the buffer when data exists in the buffer and to reflect the** non-existence of data in the buffer **when the buffer is free of data**; and

a spawner process operable to create **[a plurality of]** **the** handler processes.

2. The system of Claim 1 wherein the plurality handler processes contain a plurality of threads, wherein each thread is operable to independently handle requests.

3. The system of Claim 1 wherein the spawner process is operable to increase or decrease the number of handler processes currently in existence at any time, such operations known as load balancing.

4. The system of Claim 1 wherein the server is composed of a plurality of physical processors, each processor operable to run one or more handler processes or the spawner process.

5. A method of operating a **[true]** parallel client server system comprising **[the steps of]**:

creating a plurality of handler processes with a spawner process **at a server**;

initializing a well-known address **at the server**;

storing **[requests]** **at least one request** received by the well-known address in a buffer associated with the well-known address **at the server**;

notifying, in parallel, **[all of the]** **a** plurality **[of]** **of the** handler processes that **at least** one **[or more requests have]** **request has** arrived;

[awakening any handler processes that are currenty sleeping;]

[attempting to accept] **accepting each** pending **[requests]** **request** from the buffer, in parallel, with the plurality of handler processes **when the number of handler processes exceeds the number of pending requests**; **and**

[servicing accepted requests with those handler processes that successfully accepted a pending request; and

processing error conditions with those handler processes that did not successfully accept a pending request.]

accepting a number of pending requests substantially equal to the number of handler processes when the number of pending requests exceeds or equals the number of handler processes.

6. The method of Claim 5 wherein attempting to accept pending requests from the buffer is also performed by a plurality of threads within the plurality of handler processes.

7. The method of Claim 5 wherein creating the plurality of handler processes with the spawner process results in the plurality of processes running on a plurality of physical processors.

8. The method of Claim 5 **[wherein]** **and further comprising** increasing or decreasing the number of handler processes currently in existence with the spawner process, such operations known as load balancing.

9. The method of Claim 5 wherein the initialization of the well-known address is performed by cooperation between the operating system and the spawner process.

10. **Please cancel Claim 10 without prejudice or disclaimer**.

11. A system of operating a parallel client server system comprising:

means for creating a plurality of handler processes with a spawner process at a server;

means for initializing a well-known address at the server;

means for storing at least one request received by the well-known address in a buffer associated with the well-known address at the server;

means for notifying, in parallel, a plurality of the handler processes that at least one request has arrived;

means for accepting each pending request from the buffer, in parallel, with the plurality of handler processes when the number of handler processes exceeds the number of pending requests; and

means for accepting a number of pending requests substantially equal to the number of handler processes when the number of pending requests exceeds or equals the number of handler processes.

12. A method of operating a parallel client server system comprising the steps of:

providing, at a server, at least one available handler process, the available handler process comprising a handler process which is not presently processing a previously accepted pending request;

providing a well-known address at the server;

storing, at the server, at least one pending request received by the well-known address in a buffer associated with the well-known address;

notifying, substantially in parallel, the available handler processes that at least one pending request is in the buffer;

accepting substantially all pending requests from the buffer, substantially in parallel, with the available handler processes, when a number of pending requests is less than or equal to a number of available handler processes;

accepting a number of pending requests from the buffer substantially equal to the number of available handler processes when the number of pending requests is greater than the number of available handler processes; and

servicing accepted pending requests.

13. The method of Claim 12 and further comprising processing error conditions with those available handler processes that did not successfully accept a pending request when the number of available handler processes is greater than the number of pending requests.

14. The method of Claim 12, wherein providing the available handler processes comprises creating a plurality of the handler processes with a spawner process and wherein the available handler processes comprise a subset of the handler processes.

15. The method of Claim 14, wherein notifying comprises updating a flag and wherein the flag is accessible by substantially all the handler processes at substantially any time.

16. The method of Claim 12, wherein providing the well-known address comprises initializing the well-known address.

17. A system for a parallel client server system comprising:

software encoded on a computer readable medium, the software operable to:

provide, at a server, at least one available handler process, the available handler process comprising a handler process which is not presently processing a previously accepted pending request;

provide a well-known address at the server;

store, at the server, at least one pending request received by the well-known address in a buffer associated with the well-known address;

notify, substantially in parallel, the available handler processes that at least one pending request is in the buffer;

accept substantially all pending requests from the buffer, substantially in parallel, with the available handler processes, when a number of pending requests is less than or equal to a number of available handler processes;

accept a number of pending requests from the buffer substantially equal to the number of available handler processes when the number of pending requests is greater than the number of available handler processes; and

service accepted pending requests.

18. The system of Claim 17, wherein the software is further operable to process error conditions associated with those available handler processes that did not successfully accept a pending request when the number of available handler processes is greater than the number of pending requests.

19. The system of Claim 17, wherein the software is further operable to create a plurality of the handler processes with a spawner process and wherein the available handler processes comprise a subset of the handler processes.

20. The system of Claim 19, wherein the software is further operable to update a flag associated with the notification and wherein the flag is accessible by substantially all the handler processes at substantially any time.

21. The system of Claim 17, wherein the software is further operable to initialize the well-known address.

22. A system for a parallel client server system comprising:

means for providing, at a server, at least one available handler process, the available handler process comprising a handler process which is not presently processing a previously accepted pending request;

means for providing a well-known address at the server;

means for storing, at the server, at least one pending request received by the well-known address in a buffer associated with the well-known address;

means for notifying, substantially in parallel, the available handler processes that at least one pending request is in the buffer;

means for accepting substantially all pending requests from the buffer, substantially in parallel, with the available handler processes, when a number of pending requests is less than or equal to a number of available handler processes;

means for accepting a number of pending requests from the buffer substantially equal to the number of available handler processes when the number of pending requests is greater than the number of available handler processes; and

means for servicing accepted pending requests.

23. The method according to Claim 5 and further comprising:

servicing accepted requests with those handler processes that successfully accepted a pending request; and

processing error conditions with those handler processes that did not successfully accept a pending request.